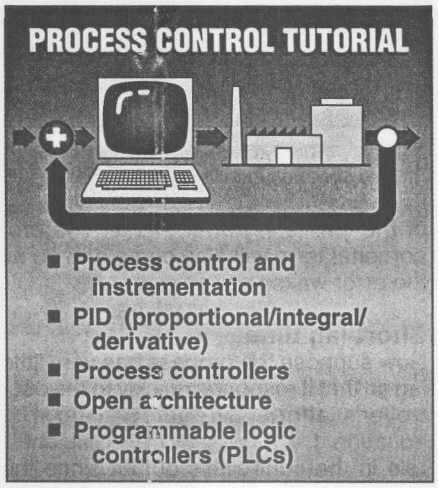
Examining the Fundamentals of PID Control

ANCE J. VANDOREN, *CONTROL ENGINEERING*

Algorithms take the simple feedback controller one step further.

This tutorial presents an overview of how and why PID controllers work. It is the first in a four part series on the fundamental concepts of modern control theory.



A feedback controller is designed to generate an "*output*" that causes some corrective effort to be applied to a "process" so as to drive a measurable "*process variable*" towards a desired value known as the "*setpoint*." Figure 1 shows a typical feedback control loop, with blocks representing the dynamic elements of the system and arrows representing the flow of information, generally in the form of electrical signals.

Virtually all feedback controllers determine their output by observing the "*error*" between the setpoint and the actual process variable measurement. A home thermostat, for example, uses the air conditioning system to correct the temperature in a process comprised of a room and the air inside. It sends an electrical signal (an output) to turn on the air conditioner when the error between the actual temperature (the process variable) and the desired temperature (the setpoint) is too high.

A look at PID control

A proportional-integral-derivative or PID controller performs much the same function as the thermostat, but with a more elaborate algorithm for determining its output. It looks at the current value of the error, the integral of the error over a recent time interval, and the current derivative of the error signal to determine not only how much of a correction to apply, but for how long. Those three quantities are each multiplied by a "tuning constant" and added together to produce the current controller output $CO(t)$, thusly:

$$CO(t)=P\cdot e(t)+I\cdot\left(\int_0^t e(\tau)d\tau\right)+D\cdot\left(\tfrac{d}{dt}e(t)\right) \qquad \text{[eq. 1]}$$

In equation [1], P is the "*proportional*" tuning constant, I is the "*integral*" tuning constant, D is the "*derivative*" tuning constant, and $e(t)$ is the error between the setpoint $SP(t)$ and the process variable $PV(t)$ at time t.

$$e(t)=SP(t)-PV(t) \qquad \text{[eq. 2]}$$

If the current error is large, has been sustained for some time, or is changing rapidly, the controller will attempt to make a large correction by generating a large output. Conversely, if the process variable has matched the setpoint for some time, the controller will leave well enough alone.

Tuning the controller

Conceptually, that's all there is to a PID controller. The tricky part is "*tuning*" it; *i.e.*, setting the P, I, and D tuning constants so that the weighted sum of the proportional, integral, and derivative terms produces a controller output that steadily drives the process variable in the direction required to eliminate the error.

The brute force solution to this problem would be to generate the largest possible output by using the largest possible tuning constants. A controller thus tuned would amplify every error and initiate extremely aggressive efforts to eliminate even the slightest discrepancy between the setpoint and the process variable. However, an overly aggressive controller can actually make matters worse by driving the process variable past the setpoint as it attempts to correct a recent error. In the worst case, the process variable will end up even further away from the setpoint than before.

On the other hand, a PID controller that is tuned too conservatively may not be able to eliminate one error before the next one appears. A well-tuned controller performs at a level somewhere

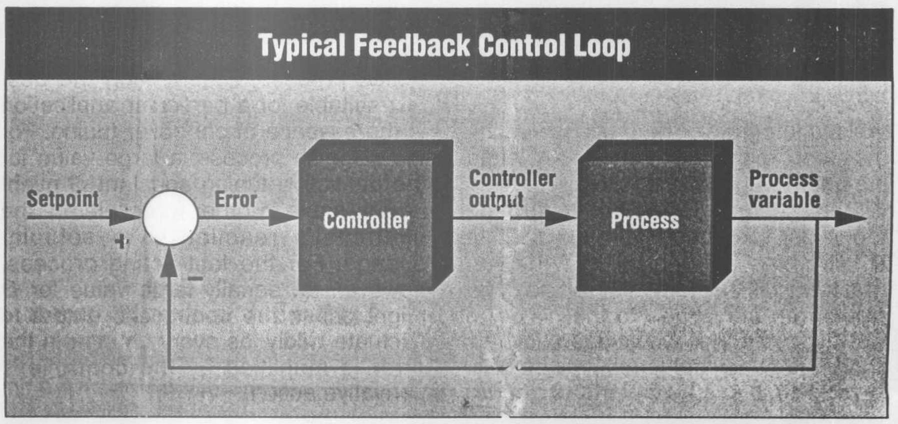


Fig. 1: Feedback controllers determine their output by observing the error between setpoint and the actual process variable measurement.

between those two extremes. It works aggressively to eliminate an error quickly, but without overdoing it.

How to best tune a PID controller depends upon how the process responds to the controller's corrective efforts. Processes that react instantaneously and predictably don't really require feedback at all. A car's headlights, for example, come on as soon as the driver hits the switch. No subsequent corrections are required to achieve the desired illumination.

On the other hand, the car's cruise controller cannot accelerate the car to the desired cruising speed as quickly. Because of friction and the car's inertia, there is always a delay between the time that the cruise controller activates the accelerator and the time that the car's speed reaches the setpoint (see Fig. 2). A PID controller must be tuned to account for such "*lags.*"

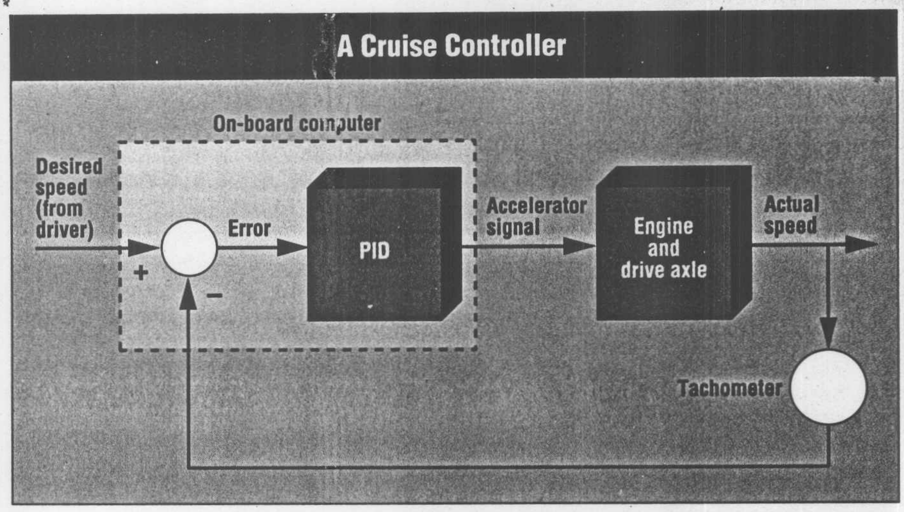


Fig. 2: A familiar real-world example of feedback control can be found in the "cruise control" feature common in many automobiles.

PID in action

Consider a sluggish process with a relatively long lag—accelerating an overloaded car with an undersized engine, for example. Such a process tends to respond slowly to the controller's efforts. If such errors are introduced abruptly (as when the setpoint is changed), the controller's initial reaction will be determined primarily by the actions of the derivative term in equation 1. This will cause the controller to initiate a burst of corrective effort the instant the error changes from zero. The proportional term will then come into play to keep the controller's output going until the error is eliminated.

After a while, the integral term will also begin to contribute to the controller's output as the error accumulates over time. In fact, the integral term will eventually come to dominate the output signal because the error decreases so slowly in a sluggish process. Even after the error has been eliminated, the controller will continue to generate an output based on the history of errors that have been accumulating in the controller's integrator. The process variable may then "overshoot" the setpoint, causing an error in the opposite direction.

If the integral tuning constant is not too large, this subsequent error will be smaller than the original, and the integral term will begin to diminish as negative errors are added to the history of positive ones. This whole operation may then repeat several times until both the error and the accumulated error are eliminated. Meanwhile, the derivative term will continue to add its share to the controller output based on the derivative of the oscillating error signal. The proportional term will also come and go as the error waxes and wanes.

Short lag time

Now suppose the process has very little lag so that it responds quickly to the controller's efforts. The integral term in equation 1 will not play as dominant a role in the controller's output since the errors will be so short-lived. On the other hand, the derivative term will tend to be larger, since the error changes rapidly in the absence of long lags.

Feedback controllers determine their output by observing the error between the setpoint and the actual process variable measurement.

Clearly, the relative importance of each term in the controller's output depends on the behavior of the controlled process. Determining the best mix suitable for a particular application is the essence of controller tuning. For the sluggish process, a large value for the derivative tuning constant *D* might be advisable in order to accelerate the controller's reaction to a setpoint change. For the fast-acting process, however, an equally large value for *D* might cause the controller's output to fluctuate wildly, as every change in the error is amplified by the controller's derivative action.

Tuning techniques

There are three schools of thought on how to select the values of *P*, *I*, and *D* required to achieve an acceptable level of performance for the controller. The first method is simple trial and error—tweak the tuning parameters and watch the controller handle the next error. If it can eliminate the error in a timely fashion, quit. If it proves to be too conservative or too aggressive, increase or decrease one or more of the tuning constants. Experienced control engineers seem to know just how much proportional, integral, and derivative action to add or subtract in order to correct the performance of a poorly tuned controller.

Unfortunately, intuitive tuning procedures can be difficult to develop because a change in one tuning constant tends to affect the performance of all three terms in the controller's output. For example, turning down the integral action reduces overshoot. This in turn slows the rate of change of the error and thus reduces the derivative action as well.

Using math models

The analytical approach to the tuning problem, which is the second method, is more rigorous. It involves a mathematical "model" of the process that relates the value of the process variable at time *t* to the current rate of change of the process variable and a history of the controller's output. For example,

$$PV(t) = K \cdot CO(t-d) - T \cdot \left(\frac{d}{dt} PV(t)\right) \qquad \text{[eq.3]}$$

This particular model describes a process with a "*gain*" of *K*, a "*time constant*" of *T*, and a "*deadtime*" of *d*. The process gain represents the magnitude of the controller's effect on the process

variable. A large value of *K* corresponds to a process that amplifies small control efforts into large changes in the process variable.

The time constant in equation 3 represents the severity of the process lag. A large value of *T* corresponds to a long lag in a sluggish process. The deadtime *d* represents another kind of delay present in many processes, where the "sensor" used to measure the process variable is located some distance from the "actuator" used to implement the controller's corrective efforts. The time required for the actuator's effects to reach the sensor is the deadtime. During that interval, the process variable does not respond at all to the actuator's activity. Only after the deadtime has elapsed does the lag time begin (see Fig. 3).

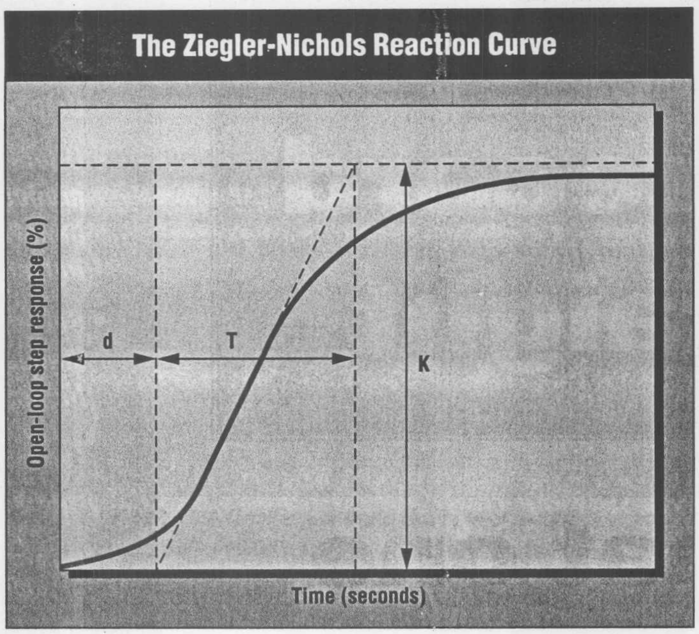


Fig. 3: Examination of the Ziegler-Nichols reaction curve reveals the response of a simple lag process to a unit step change in the controller output.

In the thermostat example above, the air conditioner is the actuator and the thermostat's onboard thermocouple is the sensor. If there is any ductwork between the air conditioner and the thermostat, there will be a deadtime while each slug of cool air travels down the duct. The room temperature will not begin to drop until the first slug of cool air emerges from the duct.

There are other characteristics of process behavior that can be factored into a process model, but equation 3 is one of the simplest and most widely used. It applies to any process with a process variable that changes in proportion to its current value. For example, a car of mass *m* accelerates when its cruise control calls for the engine to apply a force F_e to the drive axle. However, that acceleration $a(t)$ is opposed by frictional forces F_f that are proportional to the car's current velocity $v(t)$ by a factor of K_f. If the force applied by the engine is proportional to the controller's output by a factor of K_e, then applying Newton's second law to the process gives:

$$F_e - F_f = m\cdot a(t) \qquad \text{[eq. 4]}$$

$$K_e\cdot CO(t) - K_f\cdot v(t) = m\cdot\left(\tfrac{d}{dt}v(t)\right) \qquad \text{[eq. 5]}$$

$$v(t) = K\cdot CO(t) - T\cdot\tfrac{d}{dt}v(t) \qquad \text{[eq. 6]}$$

"The process variable is $v(t)$, the process gain is $K = K_e/K_f$ and the process time constant is $T = m/K_f$. In this example no deadtime exists, since the speed of the car begins to change as soon as the cruise controller activates the accelerator.

If a model like equation 3 can be defined for a process, its behavior can be quantified by analyzing the model's parameters. In equation 6, for example, the values of *K* and *T* (computed from K_e, K_f, and *m*) determine how the velocity of the car will change in response to any control effort. A model's parameters in turn dictate the tuning constants required to modify the behavior of the process with a feedback controller.

How to best tune a PID controller depends upon how the process responds to the controller's corrective efforts.

Literally hundreds of analytical techniques can translate model parameters into tuning constants. Each approach uses a different model, different controller objectives, and different mathematical tools. Several examples of analytical tuning will be explored in future installments of this series.

The Ziegler-Nichols approach

The third approach to the tuning problem is something of a compromise between purely self-teaching trial-and-error techniques and the more rigorous analytical techniques. It was originally proposed in 1942 by John G. Ziegler and Nathaniel B. Nichols, and remains popular today because of its simplicity and its applicability to any process governed by a model in the form of equation 3. Through trial-and-error experiments, Ziegler and Nichols created a set of "tuning rules" that translate the parameters of equation 3 into values for *P*, *I*, and *D*, giving generally acceptable controller performance. In particular,

$$P = \frac{1.2\cdot T}{K\cdot d}$$

$$I = \frac{0.6\cdot T}{K\cdot d^2} \qquad \text{[eqs. 7]}$$

$$D = \frac{0.6\cdot T}{K}$$

Ziegler and Nichols also came up with a practical method for estimating the values of *K*, *T*, and *d* experimentally. With the controller in manual mode (no feedback), they induced a step change in the controller's output, then analyzed the process reaction graphically (see Fig. 3). They concluded that the process gain *K* can be approximated by dividing the net change of the process variable by the size of the step change generated by the controller. They estimated the deadtime *d* from the interval between the controller's step change and the beginning of a line drawn tangent to the reaction curve at its steepest point. They also used the inverse slope of that line to estimate the time constant *T*.

Other tuning rules have since been developed for more complex models and for other controller performance objectives. Several of these, as well as a reprint of Ziegler and Nichols' 1942 paper, can be found in "Reference Guide to PID Tuning—A collection of reprinted articles of PID tuning techniques" published by *CONTROL ENGINEERING* in 1991). □

*For more information, contact Vance VanDoren Ph.D, P.E. at VanDoren Industries; Tel: 317/497-3367; Fax: 317/497-4875 or **Circle 201** on the Reader Service Card.*